**Milestone 2: R-Type Instructions**

**20% of Project Grade**

**Due: Thursday, November 9**

In this milestone, your pipeline processor need to successfully run the required R-Type MIPS instructions required in the following table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| R-Type Instruction | | | | | | |
| Instruction | OPCode | RS | RT | RD | Shamt | Funct |
| **add** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100000 |
| **addu** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100001 |
| **sub** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100010 |
| **subu** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100011 |
| **and** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100100 |
| **or** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100101 |
| **nor** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 100111 |
| **slt** $3, $2, $1 | 000000 | 00010 | 00001 | 00011 | 00000 | 101010 |
| **sll** $3, $2, 1 | 000000 | 00000 | 00010 | 00011 | 00001 | 000000 |
| **srl** $3, $2, 1 | 000000 | 00000 | 00010 | 00011 | 00001 | 000010 |
| **sra** $3, $2, 1 | 000000 | 00000 | 00010 | 00011 | 00001 | 000011 |
| **jr** $4 | 000000 | 00100 | 00000 | 00000 | 00000 | 001000 |
| **nop** | 000000 | 00000 | 00000 | 00000 | 00000 | 000000 |

* *slt rd, rs, rt*. To record the result of a less-than comparison. Compare the contents of GPR *rs* and GPR *rt* as signed integers and record the Boolean result of the comparison in GPR rd. If GPR *rs* is less than GPR *rt*, the result is 1 (true); otherwise, it is 0 (false).
* *nop*: NOP (No Operation) is the assembly idiom used to denote no operation. The actual instruction is interpreted by the hardware as SLL r0, r0, 0.
* *jr rs*, To execute a branch to an instruction address in a register. PC ← rs. Jump to the effective target address in GPR rs. Execute the instruction following the jump, in the branch delay slot, before jumping.
* *sra rd, rt, sa*, (shift word right arithmetic) To execute an arithmetic right-shift of a word by a fixed number of bits. The contents of the low-order 32-bit word of GPR *rt* are shifted right, duplicating the sign-bit (bit 31) in the emptied bits; the word result is placed in GPR *rd*. The bit-shift amount is specified by *sa*.
* *srl rd, rt, sa*, (shift right logic)To execute a logical right-shift of a word by a fixed number of bits. The contents of the low-order 32-bit word of GPR *rt* are shifted right, inserting zeros into the emptied bits; the word result is placed in GPR *rd*. The bit-shift amount is specified by *sa*.
* *sll* and *srl* (shift left logic and shift right logic). The textbook specifies these two instructions incorrectly. The textbook says R[*rd*] = R[*rs*] << *sa* or R[*rd*] = R[*rs*] >> *sa*. The MIPS standards says R[*rd*] = R[*rt*] << *sa* or R[*rd*] = R[*rt*] >> *sa*. The source register should be *rt*, instead of *rs*.

Test program 1 (It does not have any logic meaning.)

|  |
| --- |
| # Assume $1 = -30 (0xFFFFFFE2), $2 = 56 (0x00000038).  # Set $1 and $2 before running the test.  1 add $3, $2, $1 # $3 = 0x0000001a  2 sub $3, $2, $1 # $3 = 0x00000056  3 addu $3, $2, $1 # $3 = 0x0000001a  4 subu $3, $2, $1 # $3 = 0x00000056  5 and $3, $2, $1 # $3 = 0x00000020  6 or $3, $2, $1 # $3 = 0xfffffffa  7 nor $3, $2, $1 # $3 = 0x00000005  8 slt $3, $2, $1 # $3 = 0x00000000  9 slt $3, $1, $2 # $3 = 0x00000001  10 sll $3, $2, 1 # $3 = 0x00000070  11 srl $3, $2, 1 # $3 = 0x0000001c  12 sra $3, $2, 3 # $3 = 0x00000007  13 jr $2 # pc = 0x00000038  14 nop  15 nop  16 nop  17 nop |

Test program 2 (It does not have any logic meaning.)

|  |
| --- |
| # Assume $1 = -30 (0xFFFFFFE2), $2 = 56 (0x00000038).  # Set $1 and $2 before running the test.  1 add $2, $2, $1 # $2 = 0x0000001a  2 sub $2, $1, $2 # $2 = 0xffffffc8  3 addu $2, $2, $1 # $2 = 0xffffffaa  4 subu $2, $2, $1 # $2 = 0xffffffc8  5 and $2, $2, $1 # $2 = 0xffffffc0  6 or $2, $1, $2 # $2 = 0xffffffe2  7 nor $2, $2, $1 # $2 = 0x0000001d  8 slt $2, $2, $1 # $2 = 0x00000000  9 slt $2, $1, $2 # $2 = 0x00000001  10 sll $2, $2, 1 # $2 = 0x00000002  11 srl $2, $2, 1 # $2 = 0x00000001  12 sra $2, $1, 3 # $2 = 0xfffffffc  13 jr $2 # pc = 0xfffffffc  14 nop  15 nop  16 nop  17 nop |

**Milestone 2: R-Type Instructions**

**20% of Project Grade**

**TA Checkoff Sheet**

Names: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Date: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Score: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

TA Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

***Test Program 1: (50 points)***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Instruction | Address | Code | Value of $3 | Grade |
| **1** | **add** $3, $2, $1 | 0x00400000 | **0x00411820** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **2** | **sub** $3, $2, $1 | 0x00400004 | **0x00411822** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **3** | **addu** $3, $2, $1 | 0x00400008 | **0x00411821** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **4** | **subu** $3, $2, $1 | 0x0040000c | **0x00411823** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **5** | **and** $3, $2, $1 | 0x00400010 | **0x00411824** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **6** | **or** $3, $2, $1 | 0x00400014 | **0x00411825** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **7** | **nor** $3, $2, $1 | 0x00400018 | **0x00411827** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **8** | **slt** $3, $2, $1 | 0x0040001c | **0x0041182a** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **9** | **slt** $3, $1, $2 | 0x00400020 | **0x0022182a** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **10** | **sll** $3, $2, 1 | 0x00400024 | **0x00021840** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **11** | **srl** $3, $2, 1 | 0x00400028 | **0x00021842** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **12** | **sra** $3, $2, 3 | 0x0040002c | **0x000218c3** | $3 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **13** | **jr** $2 | 0x00400030 | **0x00400008** | PC = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **14** | **nop** | 0x00400034 | **0x00000000** |  | **3** |
| **15** | **nop** | 0x00400038 | **0x00000000** |  |  |
| **16** | **nop** | 0x0040003c | **0x00000000** |  |  |
| **17** | **nop** | 0x00400040 | **0x00000000** |  |  |

***Test Program 2: (50 points)***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Instruction | Address | Code | Value of $3 | Grade |
| **1** | **add** $2, $2, $1 | 0x00400000 | **0x00411020** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **2** | **sub** $2, $1, $2 | 0x00400004 | **0x00221022** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **3** | **addu** $2, $2, $1 | 0x00400008 | **0x00411021** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **4** | **subu** $2, $2, $1 | 0x0040000c | **0x00411023** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **5** | **and** $2, $2, $1 | 0x00400010 | **0x00411024** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **6** | **or** $2, $1, $2 | 0x00400014 | **0x00221025** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **7** | **nor** $2, $2, $1 | 0x00400018 | **0x00411027** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **3** |
| **8** | **slt** $2, $2, $1 | 0x0040001c | **0x0041102a** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **9** | **slt** $2, $1, $2 | 0x00400020 | **0x0022102a** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **10** | **sll** $2, $2, 1 | 0x00400024 | **0x00021040** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **11** | **srl** $2, $2, 1 | 0x00400028 | **0x00021042** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **12** | **sra** $2, $1, 3 | 0x0040002c | **0x000110c3** | $2 = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **13** | **jr** $2 | 0x00400030 | **0x00400008** | PC = \_\_\_\_\_\_\_\_\_\_\_\_\_\_ | **4** |
| **14** | **nop** | 0x00400034 | **0x00000000** |  | **3** |
| **15** | **nop** | 0x00400038 | **0x00000000** |  |  |
| **16** | **nop** | 0x0040003c | **0x00000000** |  |  |
| **17** | **nop** | 0x00400040 | **0x00000000** |  |  |